FULLY DYNAMIC HIERARCHICAL CLUSTERING FOR WELL-CLUSTERED GRAPHS

Anonymous authors
Paper under double-blind review

ABSTRACT

Hierarchical Clustering (HC) is a fundamental problem in data science. Its objective is to compute a hierarchical partition of the input data while minimizing the cost function (Dasgupta, 2016). We study the HC problem on well-clustered graphs, which admit a partition into k clusters with high inner conductance ϕ_{in}. We show that such a (k, ϕ_{in})-decomposition always exists for every connected graph, and that our assumption can be naturally derived from other notions of clusterability in the literature. We prove that a simple bucketing-and-concatenation procedure achieves an $O(k/\phi_{\text{in}})$-approximation for this problem, which becomes an $O(1)$-approximation for constant values of k and ϕ_{in}, with linear running time. Our result requires strictly weaker assumptions, and achieves a substantially better approximation ratio in less time than the previous state-of-the-art method (Laenen et al., 2023). Due to its simplicity, our algorithm is directly applicable in the fully dynamic setting and, with an update time of $O(1)$, achieves the same approximation guarantee in the static setting. The performance of our algorithm is demonstrated by experiments on both synthetic and real-world graphs.

1 INTRODUCTION

Clustering is one of the most important areas in unsupervised learning. Traditional clustering methods aim to cluster the given data points into several parts, while *hierarchical clustering* (HC) seeks to build a hierarchy of clusters for the given dataset. In recent years, HC has attracted increasing attention not only in theoretical research but also in practical applications including data science (Murtagh & Contreras, 2012), bio-informatics (Zou et al., 2021), social-networks analysis (Girvan & Newman, 2002), etc. Although the proposed hierarchical clustering methods share similar principles, they optimize different objectives in order to fit different datasets and tasks. For graphs, Dasgupta (2016) introduced a cut-based objective function to evaluate the quality of the constructed tree, and this work has inspired a number of research in graph clustering. In this paper, we study an efficient construction of the HC tree with low Dasgupta cost not only in the static setting but only in the dynamic one, where the underlying graph evolves under the edge insertions and deletions over time.

It is known that it's NP-hard to construct an HC tree with the minimum Dasgupta cost, and construct an $O(1)$-approximate tree assuming the Small Set Expansion Hypothesis (Charikar & Chatziafratis, 2017). Given these hardness results, many following works have studied the HC on specific graphs beyond the worst case, and well-clustered graphs are one of the most natural and popular assumptions. The clusterability idea is commonly used not only in graph decomposition, but also in general clustering theory (Ackerman & Ben-David, 2009; Balcan et al., 2009; 2013; Awasthi et al., 2012). In this paper, we consider the HC problem on well-clustered graphs that can be implicitly partitioned into several clusters where each cluster is internally well connected.

1.1 RELATED WORK

Dasgupta (2016) proposed an algorithm based on a recursive sparest cut algorithm to construct an HC tree, and proved that this algorithm achieves $O(\log^{3/2} n)$-approximation. Roy & Pokutta (2017) improved the approximation ratio to $O(\log n)$. Then, Charikar & Chatziafratis (2017) further improved the ratio to the state-of-the-art $O(\sqrt{\log n})$. They showed that an $O(\alpha)$-approximation algorithm for sparsest cut (e.g., Arora et al. (2004), which achieves $\alpha = O(\sqrt{\log n})$) can be used to

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

construct an $O(\alpha)$-approximate HC tree. In the fully-dynamic setting, to the best of our knowledge, there is currently no known fully dynamic HC algorithm with provable approximation guarantee specifically for the Dasgupta cost, although there are several recent methods (Monath et al., 2019; Yu et al., 2025; Rajagopalan et al., 2021) that are either heuristic or approximating other objectives.

For well-clustered graphs that admit k clusters with dense intra-cluster connections, which is the setting most relevant to our work, Manghiuc & Sun (2021) first proposed a decompose-and-bucket framework that achieves an $O(k^{22})$-approximation. Their method uses a complicated decomposition algorithm to partition the graph according to the implicit clusters, and then groups the vertices within each cluster into buckets based on their degrees and connects the buckets in a particular manner. Laenen et al. (2023) improved this approach by using spectral clustering, significantly reducing the time complexity to nearly linear while obtaining an $O(k^{23})$-approximation.

Finally, there are several other works related to the HC. Cohen-Addad et al. (2017) studied the HC problem on graphs generated by the stochastic block model (SBM) and obtained a $(1 + o(1))$-approximation in polynomial time with high probability. Other works have focused on the dual objective of the Dasgupta cost; Cohen-addad et al. (2019); Moseley & Wang (2023) obtained $O(1)$-approximation algorithms for this objective. However, although the two objectives have the same optimum, an approximation for the dual objective may yield an arbitrarily poor solution for the Dasgupta cost. More recently, Kapralov et al. (2025) proposed a sublinear-time algorithm that estimates the Dasgupta cost within an $O(\sqrt{\log k})$ factor of the optimum for well-clustered graphs. Their setting additionally assumes that the implicit clusters have low connectivity to one another, which assumption is not needed in our analysis. In addition, while their work focuses on estimating the Dasgupta cost of an HC tree, our algorithm dynamically maintains an HC tree.

1.2 Our Contribution

We propose a new hierarchical clustering framework and its dynamic implementation. The algorithm is simple, and we use Figure 1 to briefly illustrate its high-level idea. We group vertices into buckets (denoted by B_j in Figure 1) based on the powers of 2 in their degrees. We then simply construct an arbitrary tree T_{B_j} within each bucket B_j and merge these trees into a "caterpillar" tree. Surprisingly, this simple principle achieves an $O(1)$-approximation guarantee under a common clusterability assumption. Due to this simplicity, the algorithm is highly efficient, and runs in *linear* time. Our result requires a weaker assumption and achieves tighter approximation ratio with a lower running time than the previous state-of-the-art of the problem (Laenen et al., 2023).

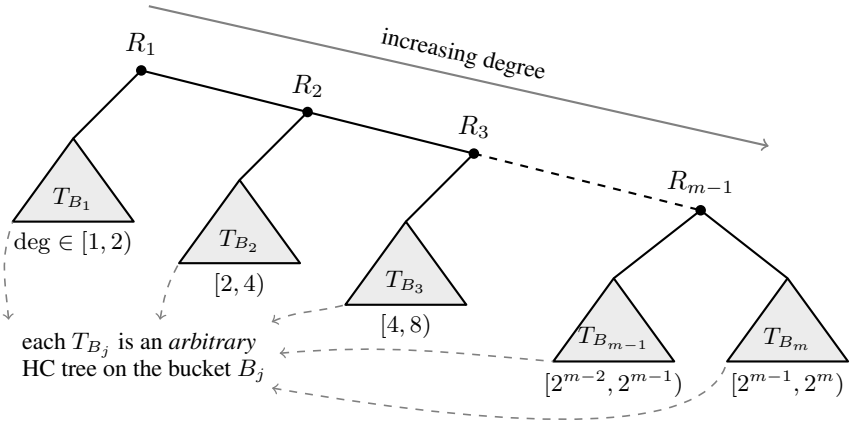


Figure 1: The structure of our constructed tree. We group the vertices into buckets according to their degrees, and organize the buckets as a caterpillar tree. Within each bucket B_j, we construct an arbitrary tree for the vertices in B_j.

Our framework is applicable in the fully dynamic setting, in which both vertices and edges can be added and deleted, and our algorithm achieves $O(1)$ update time in the worst case. We show that, in order to make our algorithm applicable in the dynamic setting, it is sufficient to (1) maintain the degree of each vertex and (2) maintain the caterpillar tree over the buckets. Both tasks can be easily

achieved in $O(1)$ time. We further conduct experiments on both synthetic and real-world datasets, and our experiments further confirm the approximation ratio and running time of the algorithm. Due to its simplicity and efficiency, we believe that our algorithm and analysis will inspire further studies in hierarchical clustering.

2 PRELIMINARIES

2.1 NOTATION

Let $G = (V, E, w)$ be the given graph with the vertex set V, edge set E, and the weight function w. We use e_{uv} to denote the edge connecting the vertices u and v with weight $w(u, v)$. When it is clear from the context, we simply use e to denote an edge and $w(e)$ to denote its weight. For two disjoint subsets $S, T \subseteq V$, their cut weight is defined as $w(S, T) \triangleq \sum_{u \in S, v \in T} w(u, v)$. For any vertex $v \in V$, the degree of v in graph G is defined as $\deg(v) \triangleq \sum_u w(u, v)$, and we assume the ratio between the maximum degree and the minimum degree of the vertices is upper bounded by a polynomial in n. For any subset $S \subseteq V$, the volume of S (in graph G) is defined by $\mathrm{vol}(S) \triangleq \sum_{v \in S} \deg(v)$, and we use $G[S]$ to denote the subgraph of G induced by S, *i.e.*, $G[S] \triangleq (V \cap S, E[S] = E \cap (S \times S), w)$. The *outer conductance* and *inner conductance* of some subset $S \subseteq V$ are defined by

$$\phi^{\mathrm{in}}(S) \triangleq \min_{\emptyset \neq T \subsetneq S} \frac{w(T, S \setminus T)}{\min\{\mathrm{vol}(T), \mathrm{vol}(S \setminus T)\}} \qquad \phi^{\mathrm{out}}(S) \triangleq \frac{w(S, V \setminus S)}{\mathrm{vol}(S)}.$$

We use $\rho(k)$ to denote the optimal k-way outer conductance of the graph G, *i.e.*,

$$\rho(k) \triangleq \min_{\substack{A_1, \ldots, A_k \subseteq V \\ \text{pairwise disjoint, non-empty}}} \max_{1 \leq i \leq k} \phi^{\mathrm{out}}(A_i). \tag{1}$$

Our analysis is based on the spectral properties of graphs, and we list the basics of spectral graph theory. For a graph $G = (V, E, w)$, let $D \in \mathbb{R}^{n \times n}$ be the degree matrix with $D(u, u) = \deg(u)$, and $A \in \mathbb{R}^{n \times n}$ the adjacency matrix with $A(u, v) = w(u, v)$. The normalized Laplacian of G is $\mathcal{L} \triangleq I - D^{-1/2} A D^{-1/2}$, where I is the identity matrix. The normalized Laplacian $\mathcal{L}$ is symmetric and positive semidefinite, with real eigenvalues $0 = \lambda_1 \leq \cdots \leq \lambda_n \leq 2$, and an associated orthonormal basis of eigenvectors $f_1, \ldots, f_n \in \mathbb{R}^n$ (Chung, 1997).

Lemma 1 (The higher-order Cheeger inequality, (Lee et al., 2014))**.** *There is an absolute constant C such that it holds for any graph G and $k \geq 2$ that $\lambda_k/2 \leq \rho(k) \leq C \cdot k^2 \sqrt{\lambda_k}$.*

2.2 HIERARCHICAL CLUSTERING

A hierarchical clustering (HC) tree of a given graph G is a binary tree T with n leaf nodes such that each leaf corresponds to exactly one vertex $v \in V$. Let T be an HC tree of a graph $G = (V, E, w)$, and $N \in T$ be an arbitrary internal node of T. We denote $T[N]$ to be the subtree of T rooted at N, $\mathsf{leaves}(T[N])$ to be the set of leaf nodes of $T[N]$. In addition, each internal node $N \in T$ induces a unique vertex set $C \subseteq V$ formed by the vertices corresponding to $\mathsf{leaves}(T[N])$. For ease of presentation, we sometimes abuse the notation and write $N \in T$ for both the internal node of T and the corresponding subset of vertices in V.

To measure the quality of an HC tree T with similarity weights, Dasgupta (2016) introduced the cost function defined by

$$\mathsf{COST}_G(T) \triangleq \sum_{e = \{u, v\} \in E} w_e \cdot |\mathsf{leaves}(T[u \vee v])|, \tag{2}$$

where $u \vee v$ is the lowest common ancestor of u and v in T. Sometimes, it is convenient to consider the cost of an edge $e = \{u, v\} \in E$ in T as $\mathsf{cost}_G(e) \triangleq w_e \cdot |\mathsf{leaves}(T[u \vee v])|$. Trees that achieve a better hierarchical clustering have a lower cost, and the objective of HC is to construct trees with the lowest cost based on the following consideration: for any pair of vertices $u, v \in V$ that correspond to an edge of high weight w_{uv}, a "good" HC tree would separate u and v lower in the tree, which is reflected in a small value of $|\mathsf{leaves}(T[u \vee v])|$. We denote by OPT_G the minimum cost of any HC tree of G, i.e., $\mathsf{OPT}_G = \min_T \mathsf{COST}_G(T)$. We say that an HC tree T is an α-approximate tree if $\mathsf{COST}_G(T) \leq \alpha \cdot \mathsf{OPT}_G$ for some $\alpha \geq 1$.

3 THE ALGORITHM IN THE STATIC SETTING

In this section we present a linear-time algorithm that constructs an HC tree for a well-clustered graph. Throughout the section, we assume that the initial input graph satisfies the following assumption, and our result is summarized in Theorem 1.

Assumption 1. *We assume the given graph G is well-clustered, i.e., there exists a k-way partition $\mathcal{S} = \{S_i\}_{i=1}^{k}$ such that it holds for all $1 \leq i \leq k$ that $\phi^{\text{in}}(S_i) \geq \phi_{\text{in}}$ for some parameter $\phi_{\text{in}} \in \mathbb{R}^{+}$.*

Theorem 1. *Let $G = (V, E, w)$ be an input graph that satisfies Assumption 1. Then, there is a linear-time algorithm that constructs an HC tree T of G satisfying*

$$\mathsf{COST}_G(T) \leq \frac{149k}{\phi_{\text{in}}}\mathsf{OPT}_G.$$

3.1 SIGNIFICANCE OF THEOREM 1

Now we discuss the significance of Theorem 1. We first claim that Assumption 1 is a natural assumption, and the proposition below shows that *every* connected graph can be decomposed so as to satisfy Assumption 1.

Proposition 1. *Let $\lambda_k > 0$ for some $k \geq 2$. Then, there exists some $1 \leq \ell \leq k-1$ satisfying the following: there exists an ℓ-partitioning of V into sets $P_1, \ldots, P_\ell$ such that it holds for each P_i that $\phi^{\text{in}}(P_i) \geq \rho(k)/(40k^2) \geq \lambda_k/(80k^2)$ and $\phi^{\text{out}}(P_i) \leq \ell\rho(\ell) = O(\ell^3)\sqrt{\lambda_\ell}$.*

Proposition 1 can be viewed as a stronger version of Theorem 1.5 of Oveis Gharan & Trevisan (2014), which uses reduced inner conductance $\phi(G[\cdot])$ rather than our ϕ_{in}. We defer the detailed proof to Appendix C. Moreover, we *do not* need to compute such a decomposition in the algorithm, only its existence is used in our analysis.

Comparing Assumption 1 with other clusterability conditions. Proposition 1 bridges Assumption 1 to other clusterability forms, and actually our assumption on ϕ_{in} can be derived from other clusterability conditions used in literature:

- If *conductance gap* $\rho_G(k+1)/\rho_G(k) > (1+\epsilon)$, then $\phi_{\text{in}} \geq \epsilon \cdot \rho(k+1)/7$.
- It holds that $\phi_{\text{in}} \geq \lambda_{k+1}/(80k^2)$ for any $k \in [n]$.

See Appendix C for formal proofs. These suggest that our assumption is weaker than the clusterability assumptions used in the literature (Kapralov et al., 2023; 2025; Laenen et al., 2023; Peng et al., 2017).

On conditions for $O(1)$-approximation. We usually assume $k = O(1)$ for the clustering task (Beideman et al., 2022; Dey et al., 2019). We highlight that, under this assumption, either one of the following conditions directly implies that our algorithm constructs an $O(1)$-approximate tree:

- $\phi_{\text{in}} = \Omega(1)$, which condition is used in Kapralov et al. (2025);
- $\lambda_{k+1} = \Omega(1)$, which condition is used in Laenen et al. (2023); Manghiuc & Sun (2021);

Comparison with Laenen et al. (2023). We further compare our work with the previous state-of-the-art (Laenen et al., 2023; Manghiuc & Sun, 2021), which assume $\lambda_{k+1} = \Omega(1)$, λ_k is sufficiently small, and obtained $O(k^{23})$-approximation. By contrast, we do not require the condition on λ_k, and achieve $O(k^3)$-approximation; our algorithm further runs much faster than theirs.

3.2 THE ALGORITHM

Our designed algorithm consists of grouping vertices into buckets, constructing a caterpillar tree for the buckets, and constructing an arbitrary tree for each bucket. Hence, we name the designed algorithm as BCA, which is described in detail as follows:

1. *Grouping vertices into buckets*. The algorithm maintains $O(\log n)$ buckets, and adds every vertex $u \in V$ into B_j if $\deg(u) \in [2^{j-1}, 2^j)$.
2. *Constructing a tree for B_j.* The algorithm constructs an arbitrary HC tree T_{B_j} for every bucket B_j.
3. *Constructing the final tree.* The final tree T is constructed as follows: (1) The root R_0 of T has two children: the bucket B_1, and the other node R_1; (2) every internal node R_i for $i \geq 1$ has two children, B_{i+1} and R_{i+1}; (3) the tree is constructed inductively as above, until all the buckets are placed in T.

See Algorithm 1 in the appendix for formal description.

3.3 Proof of Theorem 1

Now we analyze the algorithm, and prove Theorem 1. We write for any $F \subset E$ that $\mathsf{COST}_G(T, F) \triangleq \sum_{e \in F} \mathsf{cost}_G(e)$. For any node u in T, we use $L(u)$ to denote the set of leaves in the subtree rooted at u. For any $S \subset V$, let $d_{\min}(S) \triangleq \min_{u \in S} \deg(u)$, and $d_{\max}(S) \triangleq \max_{u \in S} \deg(u)$. The following lemmas will be used in our analysis.

Lemma 2. *For any partition $\mathcal{S} = \{S_1, S_2, \ldots, S_k\}$ of V, we have $\mathsf{OPT}_G \geq \sum_{i=1}^{k} \mathsf{OPT}_{G[S_i]}$.*

Lemma 3. *Let S be a subset of V with $\phi^{\text{in}}(S) \geq \phi_{\text{in}}$. Suppose $\{B_i\}_i$ is a partition of S. Then, we have $\mathsf{OPT}_{G[S]} \geq \frac{\phi_{\text{in}}}{9} \sum_{|B_i| \geq 2} d_{\min}(B_i) |B_i|^2$.*

Proof of Theorem 1. We first prove the approximation guarantee. Let T be the HC tree returned by the algorithm. Let $B_j^{(i)}$ denote the j-th cluster in a partition of S_i and $B_j = \bigcup_i B_j^{(i)}$. If $|B_j^{(i)}| = 1$, *i.e.*, $B_j^{(i)} = \{v\}$, then we have $d_{\min}(B_j^{(i)}) = \deg(v)$. Since buckets are disjoint, all buckets $B_j^{(i)}$ with size 1 contribute at most $\sum_{u \in V} \deg(u) = \mathrm{vol}(G) \leq \mathsf{OPT}_G$. Hence,

$$\begin{aligned}
\sum_{i,j} 2^j |B_j^{(i)}|^2 &\leq \sum_{i,j:|B_j^{(i)}| \geq 2} 2^j |B_j^{(i)}|^2 + \sum_{v \in B_j^{(i)} \text{ where } |B_j^{(i)}| = 1} \deg(v) \\
&\leq \sum_{i=1}^{k} \sum_{j:|B_j^{(i)}| \geq 2} 2 \cdot d_{\min}(B_j^{(i)}) |B_j^{(i)}|^2 + \mathrm{vol}(G) \qquad (3) \\
&\leq \sum_{i=1}^{k} \frac{18}{\phi_{\text{in}}} \mathsf{OPT}_{G[S_i]} + \mathsf{OPT}_G \leq \left(\frac{18}{\phi_{\text{in}}} + 1 \right) \mathsf{OPT}_G,
\end{aligned}$$

where the third inequality holds by Lemma 3 and the fourth one holds by Lemma 2. Therefore, for any fixed j we merge all $B_j^{(i)}$ to form B_j, and have that

$$\sum_j 2^j |B_j|^2 = \sum_j 2^j \left(\sum_i |B_j^{(i)}| \right)^2 \leq k \cdot \sum_{i,j} 2^j |B_j^{(i)}|^2 \leq k \left(\frac{18}{\phi_{\text{in}}} + 1 \right) \mathsf{OPT}_G,$$

where the first inequality uses the Cauchy-Schwarz inequality and the last one follows equation 3.

Now, we define $F \triangleq E \cap (\cup_j E(B_j, B_j))$, and analyze the total cost of T expressed by

$$\mathsf{COST}_G(T) = \mathsf{COST}_G(T, F) + \mathsf{COST}_G(T, E \setminus F).$$

Notice that, by our construction, every edge e in $E(B_j, B_j)$ contributes at most $w_e \cdot |B_j|$ to $\mathsf{COST}_G(T)$, so we have

$$\begin{aligned}
\mathsf{COST}_G(T, F) &\leq \sum_j |B_j| \sum_{e \in E(B_j, B_j)} w(e) \leq \sum_j |B_j| \,\mathrm{vol}(B_j) \leq \sum_j |B_j| \cdot 2 d_{\min}(B_j) |B_j| \\
&\leq \sum_j 2^j |B_j|^2. \qquad (4)
\end{aligned}$$

Next, we analyze $\mathsf{COST}_G(T, E \setminus F)$. We define $B_{\geq j} \triangleq \cup_{i \geq j} B_i$, and have that

$$\mathsf{COST}_G(T, E \setminus F) \leq \sum_j \sum_{u \in B_j} \deg(u) \, |B_{\geq j}| = \sum_j \mathrm{vol}(B_j) \, |B_{\geq j}| \leq \sum_j d_{\max}(B_j) \cdot |B_j| \cdot |B_{\geq j}|.$$

By construction, we have $d_{\max}(B_j) \leq 2^j$. Hence, it holds that

$$\begin{aligned}
\mathsf{COST}_G(T, E \setminus F) &\leq \sum_j 2^j \, |B_j| \cdot |B_{\geq j}| = \sum_j \sum_{j' \geq j} 2^j \, |B_j| \cdot |B_{j'}| \\
&= \sum_j \sum_{j' \geq j} 2^{(j-j')/2} \sqrt{2^j \, |B_j|^2} \cdot \sqrt{2^{j'} \, |B_{j'}|^2} \\
&\leq \sum_j \sum_{j' \geq j} 2^{(j-j')/2} \cdot \frac{2^j |B_j|^2 + 2^{j'} |B_{j'}|^2}{2},
\end{aligned}$$

where the last inequality follows from the AM-GM inequality. Hence,

$$\begin{aligned}
&\mathsf{COST}_G(T, E \setminus F) \leq \sum_j \left(|B_j|^2 \cdot 2^j \cdot \sum_{j' \geq j} 2^{(j-j')/2} \; + \sum_{j' \geq j} 2^{(j-j')/2} \cdot 2^{j'} \cdot |B_{j'}|^2 \right) \\
&= 2 \sum_j 2^j |B_j|^2 \left(\sum_{j' \geq j} 2^{(j-j')/2} \right) \leq 2 \sum_j 2^j \cdot |B_j|^2 \left(1 + 2^{-\frac{1}{2}} + 2^{-1} + 2^{-\frac{3}{2}} + \cdots \right) \qquad (5) \\
&\leq 2 \sum_j 2^j |B_j|^2 \cdot \frac{1}{1 - 2^{-\frac{1}{2}}} \leq 6.83 \cdot \sum_j 2^j |B_j|^2.
\end{aligned}$$

Combining equation 4 and equation 5, we have

$$\begin{aligned}
\mathsf{COST}_G(T) &= \mathsf{COST}_G(T, F) + \mathsf{COST}_G(T, E \setminus F) \\
&\leq 7.83 \sum_j 2^j |B_j|^2 \leq 7.83 \cdot k \left(\frac{18}{\phi_{\text{in}}} + 1 \right) \mathsf{OPT}(G) \leq \frac{149k}{\phi_{\text{in}}} \mathsf{OPT}(G).
\end{aligned}$$

Next, we analyze the time complexity of Algorithm 1. The algorithm takes $O(m)$ time to compute the degree of the vertices, and $O(n)$ time to group the vertices into buckets. The algorithm takes $O(|B_i|)$ time to construct an arbitrary tree inside any bucket B_i, so the overall running time for constructing the sub-trees inside all the buckets is $\sum_i |B_i| = O(n)$. Finally, building a caterpillar tree across the buckets takes $O(|\mathcal{B}|) = O(\log n)$ time, since the ratio between the maximum and minimum degrees of the vertices is upper bounded by a polynomial in n. This proves that the total time complexity of the algorithm is $O(m)$. □

4 The Algorithm in the Fully Dynamic Setting

In this section, we extend the BCA to the dynamic setting based on the following observation: (1) for every vertex u, the bucket that u belongs to is decided by $\deg(u)$ alone; (2) the tree inside a bucket is built arbitrarily, and the outside caterpillar tree is determined by the buckets themselves. Therefore, the degree information is all we need to run the BCA. Besides, Theorem 1 guarantees the output of the BCA achieves an $O(k/\phi_{\text{in}})$ approximation. This observation makes the algorithm applicable in the fully dynamic setting: it suffices to maintain the degree for each vertex and keep the buckets and tree consistent with the bucketed-caterpillar principle. Moreover, the approximation guarantee holds at every time step. This is summarized as follows:

Theorem 2. *Let G_1 be an initial graph satisfying Assumption 1. Assume that (1) edges between existing vertices can be added to or deleted from G_t at each time t, (2) isolated vertices could be added to or deleted from G_t at each time, and (3) graph G_T at every query time T satisfies Assumption 1 and has k' clusters. Then, there is a fully dynamic algorithm that maintains an $O(\frac{k'}{\phi_{\text{in}}})$-approximate solution with respect to the Dasgupta cost. The algorithm has $O(1)$ update time in the worst case.*

To describe the fully dynamic model, we use $G_t = (V_t, E_t, w)$ to denote the graph at time t, and G_t evolves over time. At every time step t, the algorithm supports the following operations:

- AddVertex(v): add a new isolated vertex v to V_t.
- DeleteVertex(v): delete v from V_t; this operation is only allowed when $\deg_t(v) = 0$.
- AddEdge(u, v, ω): insert an edge of weight $\omega > 0$.
- DeleteEdge(u, v): delete an existing edge e_{uv} from E_t.

Note that as the graph evolves, it may include isolated vertices. In the static case, we implicitly assume $\deg(v) > 0$ since isolated vertices do not contribute to the Dasgupta cost. In the dynamic setting, we can simply add one extra bucket B_0 to contain all the zero-degree vertices, and place B_0 at the top of the caterpillar. We further assume that once a new edge is added, the minimum weight of the new edge is 1, since otherwise we can simply scale all the edges. As we always assume that the maximum degree and the minimum degree of the vertices is upper bounded by a polynomial in n, scaling does not change the number of buckets.

Now we give a brief overview of the algorithm in the fully dynamic setting, and refer the reader to Algorithm 2 for the formal description. The algorithm maintains an HC tree T constructed from the initial G_0, and the degree sequence $\deg[v]$. When any of the four operations arrives, the algorithm dynamically maintains T as follows:

- the algorithm updates $\deg[v]$ for the affected vertices, and this takes $O(1)$ time;
- the algorithm moves the vertices to the right buckets due to the change of their degrees. Since we need to re-allocate at most two vertices every time step, this step takes $O(1)$ time;
- the algorithm adjusts the sub-trees corresponding to the affected buckets. Since these sub-trees are constructed arbitrarily, this step takes $O(1)$ time as well.

Remark 1. *The approximation ratio relies on the clusterability parameter k and ϕ_{in}. However, our algorithm never uses them, so we do not need to know these parameters. Additionally, Proposition 1 shows that a (k, ϕ_{in})-decomposition exists on a connected graph at any time, while isolated parts do not contribute to Dasgupta cost (equation 2). In summary, Theorem 2 holds for every time step no matter how the graph evolves.*

5 Experiments

In this section, we design experiments to compare the performance of the proposed algorithm (BCA) with the baselines including the specWRSC (Laenen et al., 2023), the AVERAGE-LINK and the LINKAGE++ (Cohen-Addad et al., 2017). All algorithms are tested on one machine with 6-core Intel Core i5-12400F at 2.5 GHz and 16 GB RAM. The datasets include synthetic data and real-world datasets. For synthetic data, we use the Stochastic Block Model (SBM) to generate well-clustered graphs. Given parameters n, k, p, q, the SBM generates a graph with n vertices and k clusters of the same size. Moreover, every pair of vertices is connected with probability p if they belong to the same cluster, and connected with probability q otherwise. By Assumption 1, we only need p to be high, and set $p = 0.15$ throughout the experiments. We fix $k = 5$ to align with previous work, and vary n from 500 to 5,000 and q from 0.10 to 0.25. For the real-world datasets, we use the WINE (Aeberhard & Forina, 1992), the CANCER (Wolberg, 1990), the MNIST (Lecun et al., 1998) and CIFAR10 (Krizhevsky, 2012). For each experiment, we repeat 10 times and report the mean and standard error bar.

5.1 Results in Static Graphs

In the static setting, we compare the BCA with the SpecWRSC (Laenen et al., 2023), the AverageLink, and the Linkage++ (Cohen-Addad et al., 2017). On synthetic datasets, we report the Dasgupta cost of each method and running time with different scale of data in Figure 2. For ease of comparison, we illustrate the cost of each method divided by that of SpecWRSC, and set the y-axis to log scale. The SpecWRSC is a constant-factor approximate algorithm for well-clustered graphs. In this experiment, the SpecWRSC achieves the lowest cost on the SBM. The SpecWRSC also uses degree-bucketing idea with two key differences: the algorithm runs spectral clustering before

bucketing and uses the brute force to build the tree over the buckets, whereas we drop the spectral clustering step and use much simpler caterpillar concatenation. The experiment shows that this simplification does not influence the approximation: with a massive speedup in runtime, the cost of the trees constructed by our algorithm never exceeds 1.15 times the one constructed by the SPECWRSC.

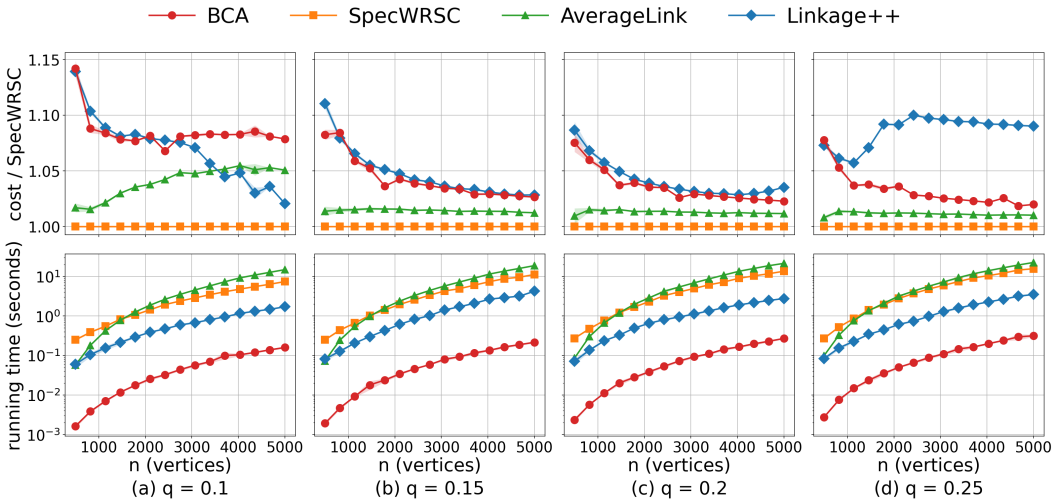


Figure 2: The Dasgupta cost and running time comparison of the four algorithms on the SBM.

On real-world datasets, we construct the similarity graph by Gaussian kernel, and set the weight of each pair of vertices x, y as $\exp(-\gamma\|x-y\|^2)$ with an independent parameter γ. Table 1 reports the Dasgupta cost divided by SPECWRSC, and Table 2 records the running time. In Table 1, the BCA beats the SPECWRSC in three datasets, while we get 2.55 times in the last WINE dataset. On running time, the BCA runs more than 10 times faster than the SPECWRSC.

Table 1: The Dasgupta Cost Comparison of the Algorithms on the Real-world Datasets

dataset	n	m	BCA	SPECWRSC	AVERAGELINK	LINKAGE++
WINE	178	11,830	2.55	1	**0.68**	3.55
CANCER	569	71,692	0.96	1	**0.61**	3.56
MNIST	1,000	499,500	0.87	1	**0.82**	0.92
CIFAR10	1,000	499,500	0.98	1	**0.87**	1.13

Table 2: Running Time Comparison of the Algorithms on the Real-world Datasets

dataset	n	m	BCA	SPECWRSC	AVERAGELINK	LINKAGE++
WINE	178	11,830	**0.007±0.001**	0.090±0.006	0.035±0.006	0.025±0.002
CANCER	569	71,692	**0.024±0.002**	0.653±0.037	0.333±0.016	0.113±0.008
MNIST	1,000	499,500	**0.131±0.011**	1.712±0.041	2.710±0.092	0.692±0.032
CIFAR10	1,000	499,500	**0.145±0.011**	1.795±0.050	2.806±0.288	0.718±0.063

5.2 RESULTS IN DYNAMIC GRAPHS

We further consider the fully dynamic setting. We consider two different dynamic operation streams, insertion and deletion. For the SBM-generated data, we insert vertices from 500 to 1,000 under different q in the insertion stream, and delete from 1,000 vertices to 500 in the deletion stream. We build the insertion/deletion stream for the four real-world datasets by the same way. Since the static baselines used before cannot fit the fully dynamic setting, we use GRINCH (Monath et al., 2019) and DYNHAC (Yu et al., 2025) as baselines, which are the latest fully dynamic hierarchical clustering methods. Our algorithm is denoted by DYNBCA. We present a portion of our experimental results here. The full results are deferred to Appendix D.

Our first dynamic experiment is on the SBM. In Figure 3, (a) and (b) are insertion streams, and (c) and (d) are deletion streams. We choose $q = 0.15$ and $q = 0.20$ for each stream (more parameter choices are deferred to Appendix D), and report the relative cost against the DynHAC and the running time. we use a logarithmic time-axis for better illustration, so the log shaped time curves actually correspond to linear growth. The experimental results show that the costs of all methods are very similar and the cost ratio never exceeds 1.03, but the running time of our algorithm is much lower than the baselines.

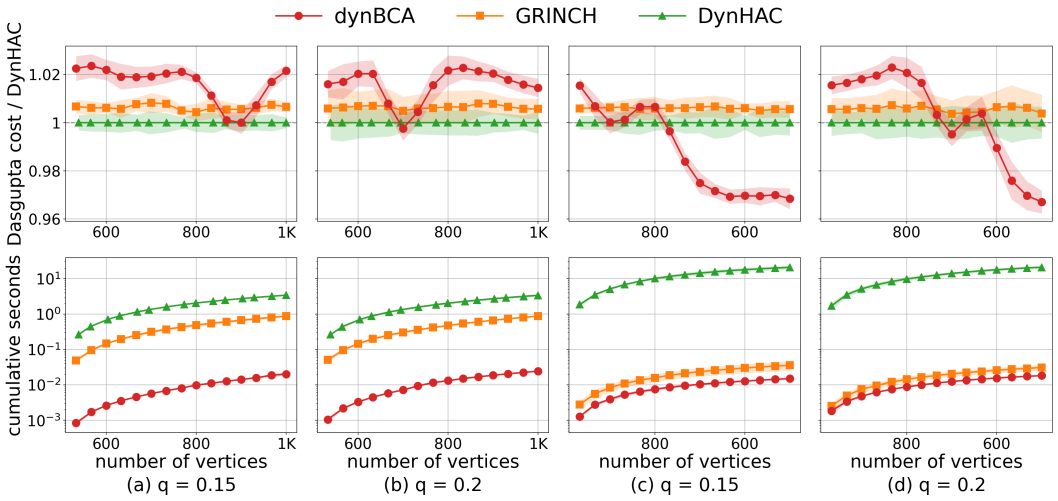


Figure 3: Dasgupta Cost and Running Time in SBM Streams

The pattern is similar on real-world datasets. We show results on WINE and CANCER (more datasets experiments are in Appendix D). (a) and (b) are insertion streams, and (c) and (d) are deletion streams. Our method obtain the lowest cost on CANCER datasets both insertion and deletion stream. On WINE, our cost is very close to the lowest curve. On all datasets, our method always wins in efficiency.

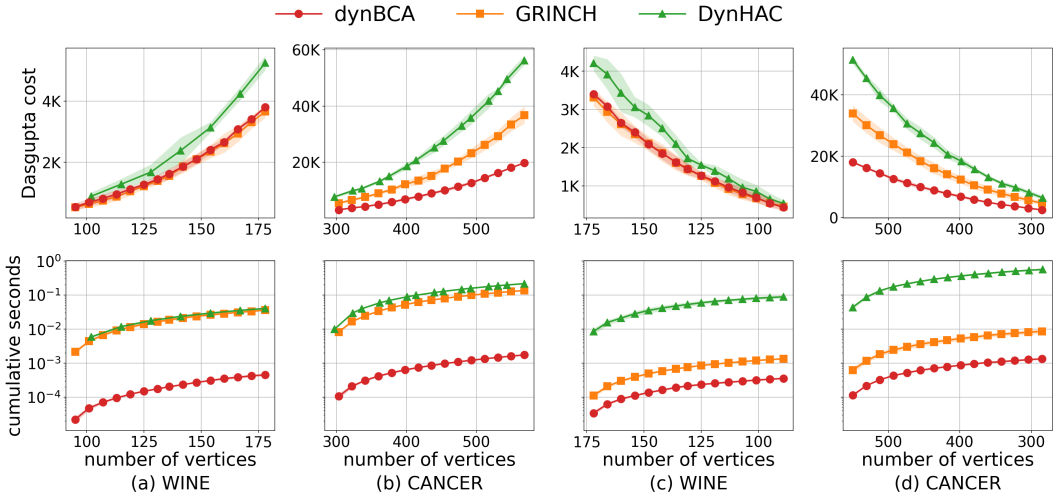


Figure 4: Dasgupta Cost and Running Time in Real-world Dataset Streams

In conclusion, our experiments confirm the excellent performance of the BCA and DynBCA algorithms shown in theory, and suggest that these algorithms could be directly applied in the real-world and dynamic settings while enjoying rigorous theoretical guarantee.

AI USE STATEMENT

In this work, we used generative AI tools for checking grammar only. We have not used generative AI tools for other tasks with required disclosure, and the rest of the required disclosure tasks are not applicable to this work. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Margareta Ackerman and Shai Ben-David. Clusterability: A theoretical study. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, pp. 1–8, 2009. URL https://proceedings.mlr.press/v5/ackerman09a.html.

Stefan Aeberhard and M. Forina. Wine. UCI Machine Learning Repository, 1992.

Sanjeev Arora, Satish Rao, and Umesh Vazirani. Expander flows, geometric embeddings and graph partitioning. In *Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing*, pp. 222–231, 2004. URL https://doi.org/10.1145/1007352.1007355.

Pranjal Awasthi, Avrim Blum, and Or Sheffet. Center-based clustering under perturbation stability. *Inf. Process. Lett.*, 112(1–2):49–54, 2012. URL https://doi.org/10.1016/j.ipl.2011.10.006.

Maria-Florina Balcan, Avrim Blum, and Anupam Gupta. Approximate clustering without the approximation. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1068–1077, 2009.

Maria-Florina Balcan, Avrim Blum, and Anupam Gupta. Clustering under approximation stability. *J. ACM*, 60(2), May 2013. URL https://doi.org/10.1145/2450142.2450144.

Calvin Beideman, Karthekeyan Chandrasekaran, and Weihang Wang. Counting and Enumerating Optimum Cut Sets for Hypergraph k-Partitioning Problems for Fixed k. In *49th International Colloquium on Automata, Languages, and Programming (ICALP 2022)*, pp. 16:1–16:18, 2022. URL https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ICALP.2022.16.

Moses Charikar and Vaggos Chatziafratis. Approximate hierarchical clustering via sparsest cut and spreading metrics. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 841–854, 2017.

Fan RK Chung. *Spectral graph theory*, volume 92. 1997.

Vincent Cohen-Addad, Varun Kanade, and Frederik Mallmann-Trenn. Hierarchical clustering beyond the worst-case. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 6202–6210, 2017.

Vincent Cohen-addad, Varun Kanade, Frederik Mallmann-trenn, and Claire Mathieu. Hierarchical clustering: Objective functions and algorithms. *J. ACM*, 66(4), 2019. URL https://doi.org/10.1145/3321386.

Sanjoy Dasgupta. A cost function for similarity-based hierarchical clustering. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*, pp. 118–127, 2016. URL https://doi.org/10.1145/2897518.2897527.

Tamal K. Dey, Pan Peng, Alfred Rossi, and Anastasios Sidiropoulos. Spectral concentration and greedy k-clustering. *Computational Geometry*, 76:19–32, 2019. URL https://www.sciencedirect.com/science/article/pii/S092577211830155X.

M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002. URL https://www.pnas.org/doi/abs/10.1073/pnas.122653799.

Michael Kapralov, Akash Kumar, Silvio Lattanzi, and Aida Mousavifar. Learning hierarchical cluster structure of graphs in sublinear time. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 925–939, 2023. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611977554.ch36.

Michael Kapralov, Akash Kumar, Silvio Lattanzi, Aida Mousavifar, and Weronika Wrzos-Kaminska. Approximating Dasgupta Cost in Sublinear Time from a Few Random Seeds. In *52nd International Colloquium on Automata, Languages, and Programming (ICALP 2025)*, pp. 103:1–103:19, 2025. URL https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ICALP.2025.103.

Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

Steinar Laenen, Bogdan-Adrian Manghiuc, and He Sun. Nearly-optimal hierarchical clustering for well-clustered graphs. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

James R. Lee, Shayan Oveis Gharan, and Luca Trevisan. Multiway spectral partitioning and higher-order cheeger inequalities. *J. ACM*, 61(6), 2014. URL https://doi.org/10.1145/2665063.

Bogdan-Adrian Manghiuc and He Sun. Hierarchical clustering: O(1)-approximation for well-clustered graphs. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, 2021.

Nicholas Monath, Ari Kobren, Akshay Krishnamurthy, Michael R. Glass, and Andrew McCallum. Scalable hierarchical clustering with tree grafting. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1438–1448, 2019. URL https://doi.org/10.1145/3292500.3330929.

Benjamin Moseley and Joshua R. Wang. Approximation bounds for hierarchical clustering: average linkage, bisecting k-means, and local search. *J. Mach. Learn. Res.*, 24(1), January 2023.

Fionn Murtagh and Pedro Contreras. Algorithms for hierarchical clustering: an overview. *WIREs Data Mining and Knowledge Discovery*, 2(1):86–97, 2012. URL https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/widm.53.

Shayan Oveis Gharan and Luca Trevisan. Partitioning into expanders. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1256–1266, 2014.

Richard Peng, He Sun, and Luca Zanetti. Partitioning well-clustered graphs: Spectral clustering works! *SIAM Journal on Computing*, 46(2):710–743, 2017. URL https://doi.org/10.1137/15M1047209.

Anand Rajagopalan, Fabio Vitale, Danny Vainstein, Gui Citovsky, Cecilia M Procopiuc, and Claudio Gentile. Hierarchical clustering of data streams: Scalable algorithms and approximation guarantees. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 8799–8809, 2021. URL https://proceedings.mlr.press/v139/rajagopalan21a.html.

Aurko Roy and Sebastian Pokutta. Hierarchical clustering via spreading metrics. *J. Mach. Learn. Res.*, 18(1):3077–3111, 2017.

WIlliam Wolberg. Breast Cancer Wisconsin (Original). UCI Machine Learning Repository, 1990. DOI: https://doi.org/10.24432/C5HP4Z.

Shangdi Yu, Laxman Dhulipala, Jakub Łącki, and Nikos Parotsidis. Dynhac: Fully dynamic approximate hierarchical agglomerative clustering. In *Proceedings of the 2025 SIAM International Conference on Data Mining (SDM)*, pp. 252–260, 2025. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611978520.24.

Ziheng Zou, Kui Hua, and Xuegong Zhang. Hgc: fast hierarchical clustering for large-scale single-cell data. *Bioinformatics*, 37(21):3964–3965, 11 2021. URL https://doi.org/10.1093/bioinformatics/btab420.

A Omitted Details from Section 3

This section presents the details omitted from Section 3.

A.1 Formal Description of the BCA Algorithm

Algorithm 1: Bucketed caterpillar HC tree

Input: Graph $G = (V, E, w)$
Output: an HC tree T of G

foreach $v \in V$ **do**
  Put v into bucket B_j, where $j = \lfloor \log \deg(v) \rfloor + 1$.
Let $B_1, \ldots, B_t$ be the non-empty buckets;
foreach $j \in [t]$ **do**
  $T_{B_j} \leftarrow$ arbitrary HC tree on B_j
$R_t \leftarrow T_{B_t}$;
for $j \leftarrow t-1$ **to** 1 **do**
  $R_j \leftarrow$ new node with left child T_{B_j} and right child R_{j+1};
return $T \leftarrow R_1$;

A.2 Omitted Proofs from Section 3

Lemma 4. *For any HC tree T of a graph G, any edge subset $F \subseteq E$, and any leaf $x \in V$, define*

$$c_F(x) \triangleq \sum_{\substack{\{u,v\} \in F \\ x \in L(u \vee v)}} w(u, v).$$

Then the following hold:

1. $\mathsf{COST}_G(T, F) = \sum_{x \in V} c_F(x)$.
2. *If u is x or an ancestor of x, then*

$$c_F(x) \geq \sum_{\substack{e = \{x,y\} \in F \\ x \in L(u), y \in V \setminus L(u)}} w_e.$$

Proof. It holds by definition that

$$\begin{aligned}\mathsf{COST}_G(T, F) &= \sum_{\{u,v\} \in F} w(u, v) |L(u \vee v)| = \sum_{\{u,v\} \in F} w(u, v) \sum_{x \in V} \mathbf{1}[x \in L(u \vee v)] \\ &= \sum_{x \in V} \sum_{\substack{\{u,v\} \in F \\ x \in L(u \vee v)}} w(u, v) = \sum_{x \in V} c_F(x).\end{aligned}$$

The second property holds because any edge of F lying in the cut $(L(u), V \setminus L(u))$ must contribute to $c_F(x)$ for $x \in L(u)$. □

Lemma 5. *For any subset $B \subseteq V$ of size $|B| \geq 2$, there is a node u in T such that $\frac{1}{3}|B| \leq |L(u) \cap B| \leq \frac{2}{3}|B|$.*

Proof. We search for the desired node u starting from the root of T as follows. When u is the root node, we have $L(u) \cap B = B$, and $|L(u) \cap B| = |B| \geq \frac{2}{3}|B|$. We pick the child u' of u such that $|L(u') \cap B| \geq |L(u) \cap B|/2$ and set $u = u'$. We continue this process until u is the first node such that $|L(u) \cap B| \leq \frac{2}{3}|B|$. Since it holds for u's parent, denoted by p, that $|L(p) \cap B| \geq \frac{2}{3}|B|$, we have $|L(u) \cap B| \geq \frac{1}{2}|L(p) \cap B| > \frac{1}{3}|B|$. □

Proof of Lemma 3. Let $\mathcal{B} = \{B_1, \ldots, B_k\}$, let T be any HC tree of $G[S]$, and let $F = E[S] = E \cap (S \times S)$ denote the induced edge set. For each $B \in \mathcal{B}$ with $|B| \geq 2$, we need to find a set $Q_B \subseteq B$ such that $|Q_B| \geq \frac{1}{3}|B|$ and $c_F(x) \geq \frac{\phi_{\text{in}}}{3} d_{\min}(B)|B|$ for all $x \in Q_B$. We construct Q_B as follows. We choose a node u by Lemma 5 and set $Q_B = L(u) \cap B$. Then, $|Q_B| \geq \frac{1}{3}|B|$ follows from Lemma 5. We have $\mathrm{vol}(L(u)) \geq \mathrm{vol}(L(u) \cap B) \geq d_{\min}(B)|L(u) \cap B| \geq \frac{1}{3} d_{\min}(B)|B|$. On the other hand,

$$\begin{aligned}\mathrm{vol}(S \setminus L(u)) &\geq \mathrm{vol}(B \setminus L(u)) \geq d_{\min}(B)|B \setminus L(u)| = d_{\min}(B)\left(|B| - |B \cap L(u)|\right) \\ &\geq \frac{1}{3} d_{\min}(B)|B|,\end{aligned}$$

where the last inequality follows from the fact that $|B \cap L(u)| \leq \frac{2}{3}|B|$. Therefore, for any $x \in Q_B$,

$$c_F(x) \geq \sum_{\substack{e=\{x,y\} \in F \\ x \in L(u), y \in V \setminus L(u)}} w_e \geq \phi_{\text{in}} \cdot \min\left\{\mathrm{vol}(L(u)), \mathrm{vol}(S \setminus L(u))\right\} \geq \frac{\phi_{\text{in}}}{3} d_{\min}(B)|B|.$$

Since $\mathcal{B}$ is a partition of S, we have

$$\mathsf{COST}_G(T, F) = \sum_{\ell \in S} c_F(\ell) \geq \sum_{\substack{B \in \mathcal{B} \\ |B| \geq 2}} \sum_{\ell \in Q_B} c_F(\ell) \geq \sum_{\substack{B \in \mathcal{B} \\ |B| \geq 2}} \frac{|B|}{3} \cdot \frac{\phi_{\text{in}}}{3} \cdot d_{\min}(B)|B|.$$

Since the analysis above holds for any tree T, the statement holds. □

Proof of Lemma 2. Let T be an optimal HC tree for G. Then, it holds for every S_i that $\mathsf{COST}_G(T, E[S_i]) \geq \mathsf{OPT}_{G[S_i]}$ by the optimality of $\mathsf{OPT}_{G[S_i]}$. Since the sets $E[S_i]$ are disjoint, the statement follows. □

B Omitted Details from Section 4

Algorithm 2: Fully Dynamic HC

Procedure Relocate(v)**:**
  $j' \leftarrow \lfloor \log \deg[v] \rfloor + 1$ if $\deg[v] > 0$, and $j' \leftarrow 0$ otherwise;
  if $j' \neq j(v)$ **then**
    unlink v from $B_{j(v)}$ **and** push v to the front of $B_{j'}$;
    $j(v) \leftarrow j'$;

Procedure AddEdge(u, v, ω)**:**
  $\deg[u] \leftarrow \deg[u] + \omega$;
  $\deg[v] \leftarrow \deg[v] + \omega$;
  Relocate(u);
  Relocate(v);

Procedure DeleteEdge(u, v)**:**
  $\omega \leftarrow w(u, v)$;
  $\deg[u] \leftarrow \deg[u] - \omega$;
  $\deg[v] \leftarrow \deg[v] - \omega$;
  Relocate(u);
  Relocate(v);

Procedure AddVertex(v)**:**
  $\deg[v] \leftarrow 0$;
  $j(v) \leftarrow 0$;
  push v to the front of B_0;

Procedure DeleteVertex(v)**:**
  // $\deg[v] = 0$, hence $v \in B_0$
  unlink v from B_0;

C Omitted Proofs from Section 3.1

We first introduce the following proposition, which is critical for proving Proposition 1.

Proposition 2 (Strengthened Existential Theorem of Oveis Gharan & Trevisan (2014)). *If $\rho_G(k+1) > (1+\epsilon)\rho_G(k)$ for some $0 < \epsilon < 1$, then the following hold:*

1. *there exist k disjoint subsets $A_1, A_2, \ldots, A_k$ such that for all $1 \le i \le k$, $\phi^{in}(A_i) \ge \epsilon \cdot \rho(k+1)/7$ and $\phi^{\mathrm{out}}(A_i) \le \rho(k)$.*
2. *there exists a k-way partition $\mathcal{S} = \{S_1, S_2, \ldots, S_k\}$ of V such that for all $1 \le i \le k$, $\phi^{\mathrm{in}}(S_i) \ge \epsilon \cdot \rho(k+1)/14k$ and $\phi^{\mathrm{out}}(S_i) \le k \cdot \rho(k)$.*

Next, we prove Proposition 2 and then Proposition 1.

Proposition 2 is the Existential Theorem 1.7 of Oveis Gharan & Trevisan (2014) with one difference: for a subset S, Oveis Gharan & Trevisan (2014) used the *induced subgraph* $G[S]$ to define the volume $\mathrm{vol}_{G[S]}$ and the inner conductance $\phi(G[S])$ of S, *i.e.*,

$$\mathrm{vol}_{G[S]}(T) = \sum_{v \in T} \deg_{G[S]}(v), \qquad \phi(G[S]) = \min_{\emptyset \neq T \subsetneq S} \frac{w(T, S \setminus T)}{\min\left\{\mathrm{vol}_{G[S]}(T), \mathrm{vol}_{G[S]}(S \setminus T)\right\}},$$

while we use the inner conductance ϕ^{in} in which volumes are measured in the whole graph G. Since $\deg_{G[S]}(v) \le \deg(v)$ for all $v \in S$ and $\mathrm{vol}_{G[S]}(T) \le \mathrm{vol}(T)$ for all $T \subseteq S$, there is a very simple relation between $\phi^{in}(S)$ and $\phi(G[S])$:

$$\begin{aligned} \phi^{in}(S) &= \min_{\emptyset \neq T \subsetneq S} \frac{w(T, S \setminus T)}{\min\left\{\mathrm{vol}(T), \mathrm{vol}(S \setminus T)\right\}} \\ &\le \min_{\emptyset \neq T \subsetneq S} \frac{w(T, S \setminus T)}{\min\left\{\mathrm{vol}_{G[S]}(T), \mathrm{vol}_{G[S]}(S \setminus T)\right\}} \;=\; \phi(G[S]). \end{aligned} \tag{6}$$

So a lower bound on ϕ^{in} is stronger than the same bound on $\phi(G[\cdot])$, and Proposition 2 does not follow from Theorem 1.7 of Oveis Gharan & Trevisan (2014) as a black box. Although most of the original proof carries over almost unchanged to our ϕ^{in} setting, we include the full proof in this appendix for completeness.

C.1 Notation and conventions

Recall the definition of $\rho(k)$ in equation 1: for any $k+1$ disjoint subsets $\mathcal{C} = \{C_1, \ldots, C_{k+1}\}$, we have

$$\max_{1 \le j \le k+1} \phi^{out}(C_j) \ge \rho(k+1). \tag{7}$$

We recall several basic properties of cut weights and volumes. For disjoint $S, T \subseteq V$, $w(S,T) = \sum_{u \in S, v \in T} w(u,v)$. Moreover, for $S \subseteq B$, we have $\mathrm{vol}(B) = \mathrm{vol}(S) + \mathrm{vol}(B \setminus S)$ and

$$w(S, V \setminus S) = w(S, B \setminus S) + w(S, V \setminus B), \qquad w(B, V \setminus B) = w(S, V \setminus B) + w(B \setminus S, V \setminus B). \tag{8}$$

For $\emptyset \neq S \subsetneq B \subseteq V$, define

$$\psi(S, B) \triangleq \frac{w(S, B \setminus S)}{\dfrac{\mathrm{vol}(B \setminus S)}{\mathrm{vol}(B)} \cdot w(S, V \setminus B)}, \qquad \Phi_B(S) \triangleq \frac{w(S, B \setminus S)}{\mathrm{vol}(S)},$$

with the convention $\psi(S, B) = +\infty$ when $w(S, V \setminus B) = 0$.

Then we have

$$\phi^{in}(B) \;=\; \min_{\emptyset \neq S \subsetneq B} \max\left\{\Phi_B(S),\ \Phi_B(B \setminus S)\right\} \;=\; \min_{\substack{\emptyset \neq S \subsetneq B \\ \mathrm{vol}(S) \le \mathrm{vol}(B)/2}} \Phi_B(S), \tag{9}$$

because $\max\{a/x, a/y\} = a/\min\{x, y\}$ and volume is additive, *i.e.*, $\mathrm{vol}(B) = \mathrm{vol}(S) + \mathrm{vol}(B \setminus S)$.

Oveis Gharan & Trevisan (2014) proposed a theoretical algorithm (Algorithm 1 in their paper) to build backbone subsets $B_1, \ldots, B_k$ from $A_1, \ldots, A_k$. The basic idea is to repeatedly cut the current set along a subset violating the second property below and keep the half with the smaller outer conductance. This procedure ensures that the outer conductance never increases, and it terminates with the claimed properties. The algorithm does not run in polynomial time, so it serves only to prove the existence of $B_1, \ldots, B_k$.

Suppose $A_1, \ldots, A_k$ are k pairwise disjoint sets achieving $\rho(k)$, i.e. $\phi^{out}(A_i) \le \rho(k)$ for all i. The assumption $\rho(k+1) > (1+\epsilon)\rho(k)$ yields $(1+\epsilon)\phi^{out}(A_i) \le \rho(k+1)$ for every i. The first lemma constructs the *backbone* B_i, whose outer conductance is no larger than that of A_i, and every subset of which sends a constant fraction of its outgoing edges back into B_i, *i.e.*, $\psi(S, B_i) \ge \epsilon/3$ for any $\emptyset \ne S \subsetneq B_i$.

Lemma 6 (Lemma 2.1 of Oveis Gharan & Trevisan (2014))**.** *Let $A_1, \ldots, A_k$ be k disjoint non-empty sets with $(1+\epsilon)\phi^{out}(A_i) \le \rho(k+1)$ for all $1 \le i \le k$. Then for every $1 \le i \le k$ there is a non-empty $B_i \subseteq A_i$ such that*

1. *$\phi^{out}(B_i) \le \phi^{out}(A_i)$, and*
2. *$\psi(S, B_i) \ge \epsilon/3$ for every $\emptyset \ne S \subsetneq B_i$.*

Oveis Gharan & Trevisan (2014) proved Lemma 6 via Algorithm 3, which repeatedly cuts the current B_i and keeps the half with the smaller outer conductance. They proved that the outer conductance never increases during the procedure and that the algorithm terminates in a finite number of steps. Since the algorithm does not run in polynomial time, Lemma 6 is not an algorithmic result, but it is sufficient for our purpose.

Algorithm 3: Construction of the backbones $B_1, \ldots, B_k$ from $A_1, \ldots, A_k$ (Oveis Gharan & Trevisan, 2014)

Input: Pairwise disjoint sets $A_1, \ldots, A_k \subseteq V$ with $(1+\epsilon)\phi^{out}(A_i) \le \rho(k+1)$
Output: Sets $B_i \subseteq A_i$ satisfying Lemma 6

for $i \leftarrow 1$ **to** k **do**
  $B_i \leftarrow A_i$;
  while *there exists* $\emptyset \ne S \subsetneq B_i$ *with* $\psi(S, B_i) \le \epsilon/3$ **do**
    $B_i \leftarrow \arg\min_{X \in \{S,\ B_i \setminus S\}} \phi^{out}(X)$;
return $B_1, \ldots, B_k$;

Lemma 7 (Lemma 2.3 of Oveis Gharan & Trevisan (2014))**.** *Let $B \subseteq V$ and $\emptyset \ne S \subsetneq B$ with $\mathrm{vol}(S) \le \mathrm{vol}(B)/2$, and write $T = B \setminus S$. If $\psi(S, B) \ge \epsilon/3$ and $\psi(T, B) \ge \epsilon/3$ for some $0 < \epsilon \le 1$, then*

$$\Phi_B(S) \;=\; \frac{w(S,T)}{\mathrm{vol}(S)} \;=\; \frac{w(S,T)}{\min\{\mathrm{vol}(S), \mathrm{vol}(T)\}} \;\ge\; \frac{\epsilon}{7} \cdot \max\left\{\phi^{out}(S), \phi^{out}(T)\right\}.$$

Proof. For $X \in \{S, T\}$, the condition $\psi(X, B) \ge \epsilon/3$ means

$$w(S,T) \;\ge\; \frac{\epsilon}{3} \cdot \frac{\mathrm{vol}(B \setminus X)}{\mathrm{vol}(B)} \cdot w(X, V \setminus B). \tag{10}$$

When $w(X, V \setminus B) > 0$, this is the definition of ψ, since $w(X, B \setminus X) = w(S, T)$. When $w(X, V \setminus B) = 0$, it holds trivially.

Taking $X = S$ in equation 10 and using $\mathrm{vol}(T)/\mathrm{vol}(B) \ge 1/2$, we have $w(S,T) \ge \frac{\epsilon}{6}\, w(S, V \setminus B)$. Adding $\frac{\epsilon}{6}\, w(S,T)$ to both sides and applying equation 8,

$$\left(1 + \frac{\epsilon}{6}\right) w(S,T) \;\ge\; \frac{\epsilon}{6}\left(w(S,T) + w(S, V \setminus B)\right) \;=\; \frac{\epsilon}{6}\, w(S, V \setminus S).$$

Dividing by $(1+\epsilon/6)\,\mathrm{vol}(S)$ and using $\epsilon \le 1$ yields $\Phi_B(S) \ge \frac{\epsilon}{6+\epsilon}\, \phi^{out}(S) \ge \frac{\epsilon}{7}\, \phi^{out}(S)$.

Now set $X = T$ in equation 10. Since $\mathrm{vol}(B) \le 2\,\mathrm{vol}(T)$, we have $\mathrm{vol}(S)/\mathrm{vol}(B) \ge \mathrm{vol}(S)/(2\,\mathrm{vol}(T))$, and dividing by $\mathrm{vol}(S)$ gives

$$\frac{w(S,T)}{\mathrm{vol}(S)} \;\ge\; \frac{\epsilon}{6} \cdot \frac{w(T, V \setminus B)}{\mathrm{vol}(T)}.$$

Adding $\frac{\epsilon}{6} \cdot \frac{w(S,T)}{\mathrm{vol}(S)}$ to both sides and then using $\mathrm{vol}(S) \le \mathrm{vol}(T)$ and equation 8 for $T \subseteq B$,

$$\left(1+\frac{\epsilon}{6}\right)\frac{w(S,T)}{\mathrm{vol}(S)} \;\ge\; \frac{\epsilon}{6}\left(\frac{w(T,V\setminus B)}{\mathrm{vol}(T)} + \frac{w(S,T)}{\mathrm{vol}(S)}\right)$$
$$\ge\; \frac{\epsilon}{6}\cdot\frac{w(T,V\setminus B)+w(T,S)}{\mathrm{vol}(T)} \;=\; \frac{\epsilon}{6}\,\phi^{out}(T).$$

Therefore, $\Phi_B(S) \ge \frac{\epsilon}{6+\epsilon}\,\phi^{out}(T) \ge \frac{\epsilon}{7}\,\phi^{out}(T)$. Combining the two bounds proves the lemma. □

Lemma 7 is the step where our statement differs from that of Oveis Gharan & Trevisan (2014). Their statement is about $\phi_{G[B_i]}(S)$, whereas we need $\Phi_B(S)$. Our statement is stronger, but nothing in the proof needs to be changed except the final conclusion.

Now we are ready to prove the first part of Proposition 2.

Proof of Proposition 2, part 1. Let $B_1, \dots, B_k$ be the sets produced by Lemma 6. They are pairwise disjoint since $B_i \subseteq A_i$, and $\phi^{out}(B_i) \le \phi^{out}(A_i) \le \rho(k) < \rho(k+1)$. For each i, let $\emptyset \ne S \subsetneq B_i$ be such that $\mathrm{vol}(S) \le \mathrm{vol}(B_i)/2$, and write $T = B_i \setminus S$. We then have the $k+1$ disjoint subsets

$$B_1, \dots, B_{i-1},\; S,\; B_i \setminus S,\; B_{i+1}, \dots, B_k.$$

By the definition of ρ in equation 7, at least one of them has outer conductance at least $\rho(k+1)$. Since $\phi^{out}(B_j) \le \rho(k) < \rho(k+1)$ for each B_j, we have $\max\{\phi^{out}(S), \phi^{out}(T)\} \ge \rho(k+1)$. By Lemma 6, we have $\psi(S, B_i) \ge \epsilon/3$ and $\psi(T, B_i) \ge \epsilon/3$. Applying Lemma 7 gives $\Phi_{B_i}(S) \ge \epsilon \cdot \rho(k+1)/7$. Combining this with equation 9, we have

$$\phi^{in}(B_i) \;\ge\; \frac{\epsilon\cdot\rho(k+1)}{7}, \qquad \phi^{out}(B_i) \;\le\; \rho(k),$$

which completes the proof. □

Now we consider the second part of Proposition 2. Since the sets $B_1, \dots, B_k$ do not necessarily form a partition, Oveis Gharan & Trevisan (2014) used a local search procedure (Algorithm 4) to assign the remaining vertices to the backbones.

Algorithm 4: Construction of the partition $P_1, \dots, P_k$ from $B_1, \dots, B_k$ (Oveis Gharan & Trevisan, 2014)

Input: Pairwise disjoint sets $B_1, \dots, B_k \subseteq V$
Output: A k-way partition $P_1, \dots, P_k$ of V with $B_i \subseteq P_i$

$P_i \leftarrow B_i$ for $1 \le i \le k-1$, and $P_k \leftarrow V \setminus (B_1 \cup \cdots \cup B_{k-1})$;
while *there exist* $i \ne j$ *and* $\emptyset \ne S \subseteq P_i \setminus B_i$ *with* $w(S, P_i \setminus S) < w(S, P_j)$ **do**
  $j^\star \leftarrow \arg\max_{j \ne i} w(S, P_j)$;
  $P_i \leftarrow P_i \setminus S$ **and** $P_{j^\star} \leftarrow P_{j^\star} \cup S$;
return $P_1, \dots, P_k$;

Lemma 8 (Fact 2.4 of Oveis Gharan & Trevisan (2014))**.** *When Algorithm 4 terminates, its output satisfies the following:*

1. *$P_1, \dots, P_k$ partition V and $B_i \subseteq P_i$ for all $1 \le i \le k$;*
2. *for every $1 \le i \le k$ and every $\emptyset \ne S \subseteq P_i \setminus B_i$, $w(S, P_i \setminus S) \ge w(S, V \setminus S)/k$.*

Oveis Gharan & Trevisan (2014) proved termination through the potential $\sum_{i<j} w(P_i, P_j)$, which strictly decreases at every iteration and takes finitely many values. Again, this algorithm need not run in polynomial time, since we only use it to prove existence.

Lemma 9 (Lemma 2.5 of Oveis Gharan & Trevisan (2014))**.** *Let $B \subseteq P \subseteq V$ and let $S = P \setminus B$. If $w(S, B) \ge w(S, V \setminus S)/k$, then $\phi^{out}(P) \le k\,\phi^{out}(B)$.*

It remains to lower-bound the inner conductance of each P_i. Fix i, write $B = B_i$ and $P = P_i$, and for $S \subseteq P$, split S and its complement in P into their backbone and leftover parts:

$$S_B \triangleq B \cap S, \qquad \overline{S}_B \triangleq B \setminus S, \qquad S_P \triangleq S \setminus B, \qquad \overline{S}_P \triangleq (P \setminus S) \setminus B.$$

Lemma 10 (Lemma 2.6 of Oveis Gharan & Trevisan (2014)). *Let $\emptyset \neq B \subseteq P \subseteq V$ and $\emptyset \neq S \subsetneq P$ such that $\mathrm{vol}(S_B) \leq \mathrm{vol}(B)/2$. Let $\rho \leq \phi^{out}(S_P)$ whenever $S_P \neq \emptyset$, and $\rho \leq \max\left\{\phi^{out}(S_B), \phi^{out}(\overline{S}_B)\right\}$ whenever $S_B \neq \emptyset$. If*

1. *$S_P \neq \emptyset$ implies $w(S_P, P \setminus S_P) \geq w(S_P, V \setminus S_P)/k$, and*
2. *$S_B \neq \emptyset$ implies $\psi(S_B, B) \geq \epsilon/3$ and $\psi(\overline{S}_B, B) \geq \epsilon/3$,*

then $\Phi_P(S) = \dfrac{w(S, P \setminus S)}{\mathrm{vol}(S)} \geq \dfrac{\epsilon \cdot \rho}{14k}$.

Proof. We split $S = S_B \cup S_P$ and $P \setminus S = \overline{S}_B \cup \overline{S}_P$. Every edge from S_P to $P \setminus S$ and every edge from S_B to $\overline{S}_B$ crosses from S to $P \setminus S$, so

$$w(S, P \setminus S) \;\geq\; w(S_P, P \setminus S) + w(S_B, \overline{S}_B). \tag{11}$$

Moreover $\overline{S}_B \neq \emptyset$, because $B \neq \emptyset$ and $\mathrm{vol}(S_B) \leq \mathrm{vol}(B)/2 < \mathrm{vol}(B)$. We consider two cases.

Case 1: $\mathrm{vol}(S_B) \geq \mathrm{vol}(S_P)$. Then $S_B \neq \emptyset$: otherwise $\mathrm{vol}(S_P) \leq 0$ forces $S_P = \emptyset$ and hence $S = \emptyset$. So S_B is a non-empty subset of B with $\mathrm{vol}(S_B) \leq \mathrm{vol}(B)/2$. Using equation 11 and $\mathrm{vol}(S) = \mathrm{vol}(S_B) + \mathrm{vol}(S_P) \leq 2\,\mathrm{vol}(S_B)$,

$$\Phi_P(S) \;\geq\; \frac{w(S_B, \overline{S}_B)}{2\,\mathrm{vol}(S_B)} \;=\; \frac{\Phi_B(S_B)}{2} \;\geq\; \frac{\epsilon}{14} \max\left\{\phi^{out}(S_B), \phi^{out}(\overline{S}_B)\right\} \;\geq\; \frac{\epsilon \cdot \rho}{14} \;\geq\; \frac{\epsilon \cdot \rho}{14k},$$

where the second inequality holds because of Lemma 7.

Case 2: $\mathrm{vol}(S_P) > \mathrm{vol}(S_B)$. Then $S_P \neq \emptyset$ and $\mathrm{vol}(S) \leq 2\,\mathrm{vol}(S_P)$. We first show

$$w(S_B, \overline{S}_B) \;\geq\; \frac{\epsilon}{6}\, w(S_B, S_P). \tag{12}$$

If $S_B = \emptyset$, both sides are zero. Otherwise, the second condition implies that $\psi(S_B, B) \geq \epsilon/3$, which as in equation 10 unfolds to $w(S_B, \overline{S}_B) \geq \frac{\epsilon}{3} \cdot \frac{\mathrm{vol}(\overline{S}_B)}{\mathrm{vol}(B)}\, w(S_B, V \setminus B)$. Here $\mathrm{vol}(\overline{S}_B) \geq \mathrm{vol}(B)/2$, and $w(S_B, V \setminus B) \geq w(S_B, S_P)$ because $S_P \subseteq V \setminus B$. This proves equation 12.

Combining equation 12 and equation 11, we have

$$\begin{aligned} w(S, P \setminus S) \;&\geq\; w(S_P, P \setminus S) + \frac{\epsilon}{6}\, w(S_P, S_B) \\ &\geq\; \frac{\epsilon}{6}\left(w(S_P, P \setminus S) + w(S_P, S_B)\right) \;=\; \frac{\epsilon}{6}\, w(S_P, P \setminus S_P), \end{aligned}$$

where the second inequality holds because $\epsilon/6 \leq 1$, and the last equality holds because $P \setminus S_P$ is the union of $P \setminus S$ and S_B. By the first condition, $w(S_P, P \setminus S_P) \geq w(S_P, V \setminus S_P)/k$. Therefore,

$$\Phi_P(S) \;\geq\; \frac{\epsilon \cdot w(S_P, V \setminus S_P)}{6k \cdot 2\,\mathrm{vol}(S_P)} \;=\; \frac{\epsilon}{12k}\, \phi^{out}(S_P) \;\geq\; \frac{\epsilon \cdot \rho}{12k} \;\geq\; \frac{\epsilon \cdot \rho}{14k}.$$

In both cases $\Phi_P(S) \geq \epsilon\rho/(14k)$. □

Proof of Proposition 2, part 2. Let $B_1, \ldots, B_k$ be the output of Algorithm 3 and let $P_1, \ldots, P_k$ be the output of Algorithm 4; by Lemma 8 $P_1, \ldots, P_k$ form a k-way partition of V with $B_i \subseteq P_i$.

Outer conductance. Applying Lemma 9 with $B = B_i$ and $P = P_i$ directly yields $\phi^{out}(P_i) \leq k\,\phi^{out}(B_i) \leq k\,\rho(k)$.

Inner conductance. Fix i and let $\emptyset \neq S \subsetneq P_i$. Since $\mathrm{vol}(S_B) + \mathrm{vol}(\overline{S}_B) = \mathrm{vol}(B_i)$ and the ratio $w(S, P_i \setminus S)/\min\{\mathrm{vol}(S), \mathrm{vol}(P_i \setminus S)\}$ is unchanged when S is exchanged with $P_i \setminus S$, we may assume $\mathrm{vol}(S_B) \leq \mathrm{vol}(B_i)/2$. Before applying Lemma 10, we check its two conditions: (1) the first condition is satisfied by property 2 of Lemma 8, since $S_P \subseteq P_i \setminus B_i$; and (2) the second condition is satisfied by Lemma 6. Therefore, Lemma 10 yields $w(S, P_i \setminus S)/\mathrm{vol}(S) \geq \epsilon \cdot \rho(k+1)/(14k)$, and since $\min\{\mathrm{vol}(S), \mathrm{vol}(P_i \setminus S)\} \leq \mathrm{vol}(S)$,

$$\frac{w(S, P_i \setminus S)}{\min\{\mathrm{vol}(S), \mathrm{vol}(P_i \setminus S)\}} \;\geq\; \frac{w(S, P_i \setminus S)}{\mathrm{vol}(S)} \;\geq\; \frac{\epsilon \cdot \rho(k+1)}{14k}.$$

Taking the minimum over all $\emptyset \neq S \subsetneq P_i$ gives $\phi^{in}(P_i) \geq \epsilon \cdot \rho(k+1)/(14k)$, which completes the proof. □

Let A be the weighted adjacency matrix of G, *i.e.*, $A_{uv} = w(u,v)$. Let $\mathcal{L}_G$ denote the normalized Laplacian of graph G, *i.e.*, $\mathcal{L}_G = I - D^{-1/2}AD^{-1/2}$, where D is the degree diagonal matrix, *i.e.*, $D = \mathrm{diag}(\deg(v))_{v\in V}$. The eigenvalues of $\mathcal{L}_G$ are denoted by $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$. For a mapping $f : V \to \mathbb{R}$, the Rayleigh quotient is defined by

$$\mathcal{R}(f) \triangleq \frac{\sum_{e_{uv}\in E} w(u,v)\,(f(u)-f(v))^2}{\sum_{v\in V}\deg(v)\,f(v)^2}. \tag{13}$$

Recall from equation 1 that $\rho(k)$ is the minimum of $\max_i \phi^{out}(A_i)$ over all families of k pairwise disjoint non-empty subsets $A_1, \ldots, A_k$ of V. We assume $|V| \geq k+1$, so that $\rho(k+1)$ is defined, and that G has no isolated vertices. For a non-empty subset $A_i \subseteq V$, we define its indicator $g_i = \mathbf{1}_{A_i}$, *i.e.*, $g_i(v) = 1$ if $v \in A_i$ and $g_i(v) = 0$ otherwise. So We have

$$\mathcal{R}(g_i) = \phi^{out}(A_i). \tag{14}$$

Lemma 11. *Let $k \geq 1$ and let $A_1, \ldots, A_k$ be pairwise disjoint non-empty subsets of V, not necessarily covering V. Then*

$$\lambda_k \;\leq\; 2 \max_{1\leq i\leq k} \phi^{out}(A_i).$$

Proof of Lemma 11. Let $x_i \triangleq D^{1/2}g_i$. Each x_i is non-zero because $A_i \neq \emptyset$ and every degree is positive. Since the sets A_i are disjoint, the vectors x_i have disjoint supports and are therefore orthogonal. Hence, $U \triangleq \mathrm{span}\,\{x_1, \ldots, x_k\}$ has $\dim U = k$, and $\{x_1, \ldots, x_k\}$ forms a basis of U. So every non-zero vector $x \in U$ can be written as $x = \sum_{i=1}^k c_i x_i$. Since G has no isolated vertices, $D^{1/2}$ is invertible, so we can further define $f = D^{-1/2}x = \sum_{i=1}^k c_i \mathbf{1}_{A_i}$. For each $v \in V$, $f(v) = c_i$ if $v \in A_i$, and $f(v) = 0$ if $v \notin A_1 \cup \cdots \cup A_k$. So we have

$$x^\top x \;=\; \sum_{v\in V}\deg(v) f(v)^2 \;=\; \sum_{i=1}^k c_i^2\,\mathrm{vol}(A_i). \tag{15}$$

Now we bound $x^\top \mathcal{L}_G x$, and have that

$$\begin{aligned}
x^\top \mathcal{L}_G x &= \sum_{e_{uv}\in E} w(u,v)\,(f(u)-f(v))^2 \\
&\leq \sum_{i=1}^k 2c_i^2\, w(A_i, V\setminus A_i) \\
&= 2\sum_{i=1}^k c_i^2\,\phi^{out}(A_i)\,\mathrm{vol}(A_i) \\
&\leq 2\left(\max_{1\leq i\leq k}\phi^{out}(A_i)\right)\sum_{i=1}^k c_i^2\,\mathrm{vol}(A_i) \\
&= 2\left(\max_{1\leq i\leq k}\phi^{out}(A_i)\right)\cdot x^\top x,
\end{aligned} \tag{16}$$

where the last equality follows from equation 15. For the first inequality, an edge e_{uv} with $f(u) = f(v)$ contributes nothing; in particular, edges inside a single A_i and edges with both endpoints outside $A_1 \cup \cdots \cup A_k$ contribute nothing. For each remaining edge we use $(f(u)-f(v))^2 \leq 2f(u)^2 + 2f(v)^2$ and charge the two terms separately. If $f(u) \neq 0$, then u lies in exactly one A_i and $f(u)^2 = c_i^2$. Since e_{uv} is not internal to A_i, we have $v \notin A_i$, so e_{uv} lies in the cut $(A_i, V\setminus A_i)$. The same holds with u and v swapped. Thus each edge of the cut $(A_i, V\setminus A_i)$ receives at most $2c_i^2$ times its weight, and summing over the cuts yields the bound.

Finally, by the Courant–Fischer min-max theorem, we have

$$\lambda_k \;=\; \min_{\substack{U'\subseteq\mathbb{R}^V \\ \dim U' = k}} \max_{0\neq x\in U'} \frac{x^\top\mathcal{L}_G x}{x^\top x} \;\leq\; \max_{0\neq x\in U}\frac{x^\top\mathcal{L}_G x}{x^\top x} \;\leq\; 2\max_{1\leq i\leq k}\phi^{out}(A_i). \qquad \square$$

We follow the proof of Theorem 1.5 of Oveis Gharan & Trevisan (2014) to prove Proposition 1, using Proposition 2 in place of their Theorem 1.7. Because Proposition 2 already bounds ϕ^{in}, the conclusion holds for ϕ^{in}. As before, we assume that G is connected, and in particular has no isolated vertices. We will prove that there is an $\ell \in \{1, \ldots, k-1\}$ and an ℓ-way partition $P_1, \ldots, P_\ell$ of V such that, for all $1 \le i \le \ell$,

$$\phi^{in}(P_i) \;\ge\; \frac{\rho(k)}{40k^2} \;\ge\; \frac{\lambda_k}{80k^2}, \qquad \phi^{out}(P_i) \;\le\; \ell\,\rho(\ell) \;=\; O(\ell^3)\sqrt{\lambda_\ell}. \tag{17}$$

Proof of Proposition 1. Assume $\lambda_k > 0$ for some $k \ge 2$. By Lemma 11, $\lambda_k \le 2\rho(k)$, and hence $\rho(k) > 0$. Since $\rho(1) = 0$, we have $\rho(\ell+1) > (1 + 1/k)\,\rho(\ell)$ for at least one index $1 \le \ell \le k-1$. Let ℓ be the largest such index. It follows that

$$\rho(k) \;\le\; \left(1 + \frac{1}{k}\right)^{k-\ell-1} \rho(\ell+1) \;\le\; \left(1 + \frac{1}{k}\right)^{k} \rho(\ell+1) \;\le\; e \cdot \rho(\ell+1). \tag{18}$$

Then, applying Proposition 2 with $\epsilon = 1/k$, we obtain an ℓ-way partition $P_1, \ldots, P_\ell$ of V such that for all $1 \le i \le \ell$,

$$\phi^{in}(P_i) \;\ge\; \frac{\epsilon \cdot \rho(\ell+1)}{14\ell} \;=\; \frac{\rho(\ell+1)}{14k\ell} \;\ge\; \frac{\rho(k)}{14e \cdot k\ell} \;\ge\; \frac{\rho(k)}{40k^2} \;\ge\; \frac{\lambda_k}{80k^2}.$$

Finally, the higher-order Cheeger inequality $\rho(\ell) \le O(\ell^2)\sqrt{\lambda_\ell}$ of Lee et al. (2014) gives $\ell\,\rho(\ell) = O(\ell^3)\sqrt{\lambda_\ell}$, which completes equation 17. □

D More Experiments

In this appendix, we add more details and results about our experiments.

D.1 Datasets

Stochastic block model. The stochastic block model has four parameters n, k, p, q: n is the total number of vertices, k is the number of blocks, and p and q control the probability of edge generation. In the stochastic block model, each block has approximately n/k vertices. Every pair of vertices is connected with probability p if they belong to the same cluster, and connected with probability q otherwise.

Table 3: SBM settings. In both experiments the $k = 5$ blocks have equal size and $p = 0.15$.

	static (Figure 2)	dynamic (Figure 3)
q	0.10, 0.15, 0.20, 0.25	0.10, 0.15, 0.20, 0.25
n	15 sizes from 500 to 5,000	1,000
m	from 14,000 to 2.9 million	55,000 to 115,000
stream	–	insert or delete 500 vertices, one at a time

Real-world datasets. To evaluate our algorithm on real-world data of different sizes and dimensions, we use four datasets, summarized below and in Table 4:

- **Wine** (Aeberhard & Forina, 1992): A UCI dataset of 178 wines grown in the same region of Italy and derived from three cultivars, each described by 13 chemical constituents (13 features).
- **Cancer** (Wolberg, 1990): The Wisconsin diagnostic breast cancer dataset from UCI, containing 569 samples labelled benign or malignant, each described by 30 features computed from a digitized image of a fine needle aspirate of a breast mass.
- **MNIST** (Lecun et al., 1998): A collection of 70,000 grayscale images of handwritten digits (0–9) of size 28×28 pixels (784 features). We use the 10,000 images of the test split.

- **CIFAR-10** (Krizhevsky, 2012): A collection of 60,000 color images of 10 object classes of size 32×32 pixels with 3 color channels (3,072 features). We use the 10,000 images of the test split.

For the static experiments, we use the similarity graph generated by a Gaussian kernel. For each pair $x, y \in V$, the weight between them is set to $\exp(-\gamma\|x-y\|^2)$, and edges of weight below 10^{-10} are dropped. Since the graph is dense, we sample 1,000 vertices from each of the large datasets, *i.e.*, MNIST and CIFAR10.

Table 4: Real-world datasets. n is the number of points in a graph and d the number of features. m is the number of edges.

dataset	points	d	classes	n	k	m static	m dynamic
WINE	178	13	3	178	5	11,830	1,231
CANCER	569	30	2	569	5	71,692	4,277
MNIST	10,000	784	10	1,000	10	499,500	7,835
CIFAR10	10,000	3,072	10	1,000	10	499,500	9,218

D.2 Additional Figures

In this section, we present additional results on the performance of our dynamic algorithm and the baselines on insertion and deletion streams.

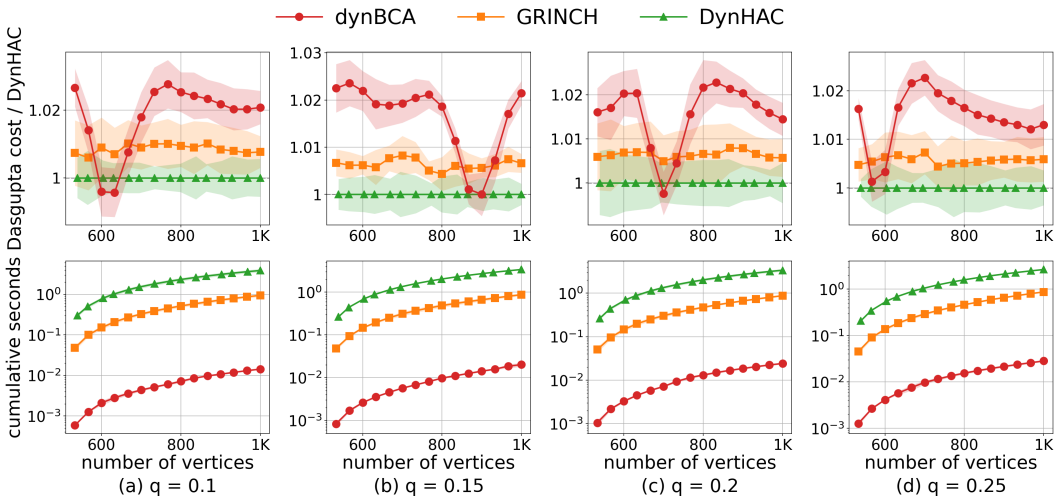


Figure 5: Dasgupta Cost and Running Time in SBM Insertion Stream

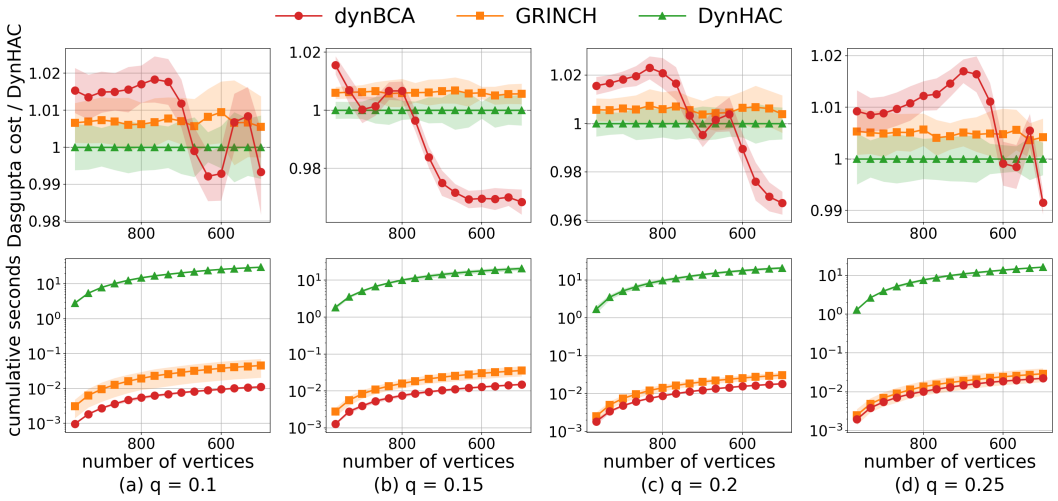


Figure 6: Dasgupta Cost and Running Time in SBM Deletion Stream

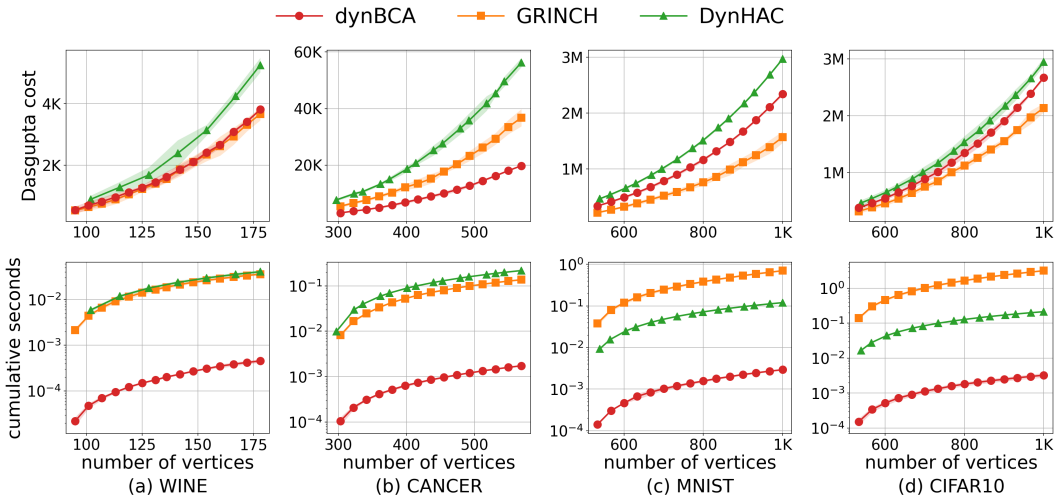


Figure 7: Dasgupta Cost and Running Time in Real-world Data Insertion Stream

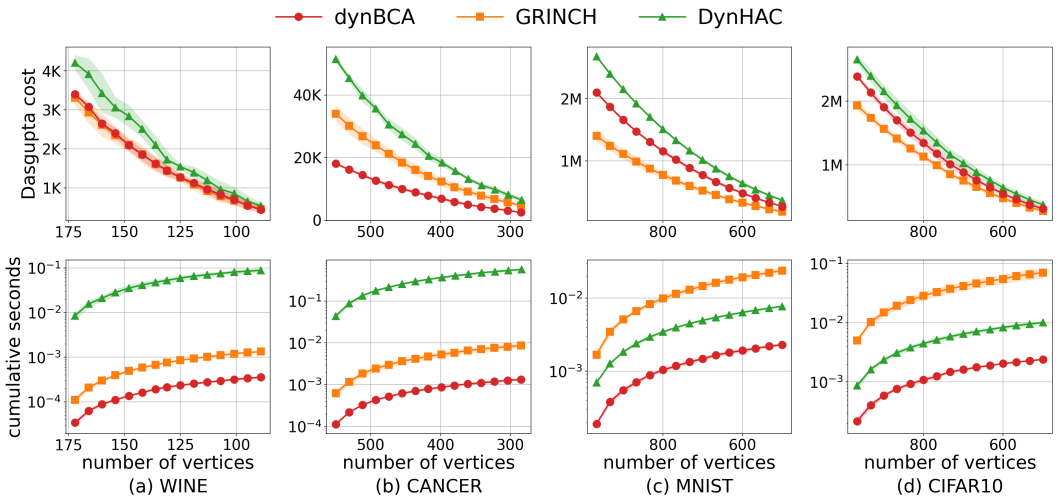


Figure 8: Dasgupta Cost and Running Time in Real-world Data Deletion Stream